Worst Case Optimal Joins on Relational and XML data

Chen, Yuxing

2018-06-10

Chen , Y 2018 , Worst Case Optimal Joins on Relational and XML data . in SIGMOD '18 Proceedings of the 2018 International Conference on Management of Data . pp. 1833-1835 , ACM SIGMOD/PODS International Conference on Management of Data , Houston, TX , United States , 10/06/2017 . https://doi.org/10.1145/3183713.3183721

http://hdl.handle.net/10138/307260
https://doi.org/10.1145/3183713.3183721

unspecified
acceptedVersion

Worst Case Optimal Joins on Relational and XML data

Yuxing Chen
University of Helsinki
yuxing.chen@helsinki.fi

ABSTRACT

In recent data management ecosystem, one of the greatest challenges is the data variety. Data varies in multiple formats such as relational and (semi-)structured data. Traditional database handles a single type of data format and thus its ability to deal with different types of data formats is limited. To overcome such limitation, we propose a multi-model processing framework for relational and semi-structured data (i.e. XML), and design a worst-case optimal join algorithm. The salient feature of our algorithm is that it can guarantee that the intermediate results are no larger than the worst-case join results. Preliminary results show that our multi-model algorithm significantly outperforms the baseline join methods in terms of running time and intermediate result size.

1 MOTIVATION

As more businesses realized that data, in all forms and sizes, is critical to making the best possible decisions, we see the continued growth of demands to manage and process massive volume of different types of data [4]. The data presents in various types and formats: structured, semi-structured and unstructured. A traditional DB typically handles only one kind of data format. Relational DB, for example, can only deal with relational tables. It is promising to develop a multi-model database to manage and process multiple data models against a single model, while meeting the increasing requirements for scalability and performance [4, 6]. In this paper, we investigate the case of the join between relational database and XML database. We show a simple example in Figure 1. Our multi-model framework computes XML twigs into a relational-like structures without losing size bound so that the worst-case size bound of join between Relational and XML can be calculated. Based on the worst-case size bound, we design a worst-case optimal join algorithm for relational and XML data.

2 RELATED WORK

Relational joins are the core operation in relational algebra as well as the core query in a standard database. Thanks to AGM bound [2], we can optimize the relational joins based on the worst-case size bound. Several optimal algorithms have proposed accordingly. For example, Ngo et al. [7] proposes an optimal algorithm for a general case and later provides a survey [8] to seek a better solution than the worst-case algorithm; Veldhuizen [9] proposes an optimal algorithm Called Leapfrog which is simple to implement. XML trees, different from relational, use twig pattern as XML queries to find all the occurrences in an XML DB. Optimal solutions (e.g., stack-tree [1], TJFast [5]) have proposed to match solution for some limited cases such as optimal match in twig ancestor-descendant relationship but not in twig child-parent relationship. Previous works of XML twig queries do not consider the worst-case size bound at all. Both Relational and XML works consider only single type of data format. Therefore, we propose a worst-cast optimal join solution in multiple data formats, which in our case are Relational and XML. Unlike traditional database management system which typically handles one single data model in data organization, storage, and manipulation, a multi-model database [3] supports multiple data models that can simultaneously process and integrate the data in different formats, instead of combining intermediate results from multiple DBs in a relatively high level way. For example, an intersection results need to be calculated from XML trees and relational tables. A multi-model processing method is to retrieve data from both DBs at the same time by considering the data size of the intermediate result, rather than naively combining both direct query results. The latter naive way normally does not satisfy the worst-case size bound when considering as a whole system.

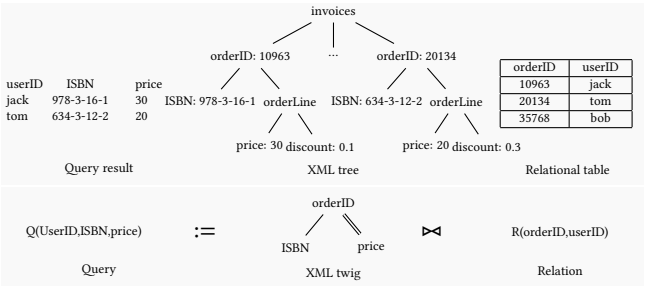

Figure 1: An example of join between XML and Relational

3 APPROACH

In XML, the terms parent, child, ancestor, and descendant are used to describe the relationships between elements. A parent-child (P-C) relationship is a direct connection between two elements. A ancestor-descendant (A-D) relationship is a single- or multi-level P-C connection between two elements.

An XML twig query can be rewritten into multiple relational-like relations without losing the worst-case size bound. We then compute the size bound for XML twig as we compute for relational joins. In terms of worst-case size bound, we transform XML twig into relational tables: (1) for each A-D edges, split the ancestor part and the descendant part into different sub-twigs; (2) for all sub-twigs, fetch all root-leaf paths; (3) for each root-leaf path (i.e. continuous P-C relationship), consider as a relational table.

SIGMOD'18, Houston, TX, USA
 ISBN 978-1-4503-4703-7/18/06…$15.00
DOI: 10.1145/3183713.3183721

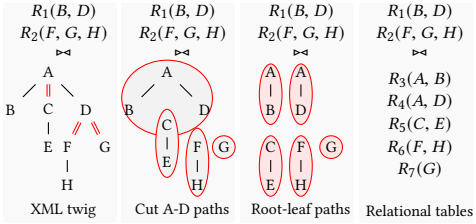

Figure 2: An example of computing worst-case size bound for Relational and XML twig

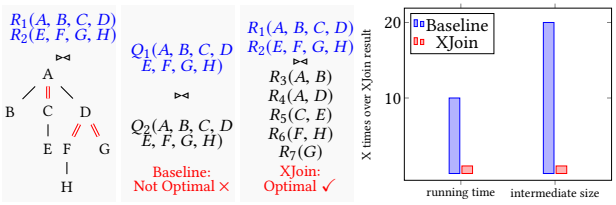

Figure 3: An example of optimally join XML twig and relational tables

Size bound for multi-model join: Fractional edge cover is originally proposed for relational size bound in AGM bound [2]. The size bound of multi-model framework is computed by the dual formulation, which is described as follows. Given relational tables R and XML twigs X. Let A_R be the set of attributes of R and A_X be the set of attributes of X. Let R_X be the transformed relational-like tables of X. By the Linear program duality, there is a solution $(y_a : a \in A_R \cup A_X)$ for the dual linear program:

$$\begin{aligned} \text{maximize} \quad & \sum_{a}^{A_R \cup A_X} y_a \\ \text{subject to} \quad & \sum_{a}^{A_R \cup A_X} y_a \leq 1 \\ & y_a \geq 0 \qquad \text{for all } a \in A_R \cup A_X \end{aligned} \tag{1}$$

Lemma 3.1. *Let $A_R \cup A_X$ be the union of attributes of R and X, and A' be the attributes from a join query Q for Relational and XML. Let D be an instance, then for join query Q, the following property holds: $\prod_{a'}^{A'} |Q(D)|^{y_{a'}} \leq \prod_{a}^{A_R \cup A_X} |R(D)|^{y_a}$*

Lemma 3.2. *Let $A_R \cup A_X$ be the union of attributes of R and X, and A' be the attributes from a join query Q for Relational and XML. For every $N_0 \in \mathbb{N}$ there is a instance D such that $N_0 \leq |D|$ and $\prod_{a'}^{A'} |Q(D)|^{y_{a'}} \geq \prod_{a}^{A_R \cup A_X} |R(D)|^{y_a}$*

Example 3.3. Consider the example in Figure 2. Given relational tables $R_1(B, D)$, $R_2(F, G, H)$ and twig $X(A, B, C, D, E, D, F, G, H)$, the goal is to find the size bound of query $Q(A, B, C, D, E, D, F, G, H)$ for joining R_1, R_2 and X. We firstly cut the A-D edges and get sub-twigs. From sub-twigs, we get all the root-leaf paths which will consider as relational tables. The equivalent relations are $R_4(A, B)$, $R_4(A, D)$, $R_5(C.E)$, $R_6(F, H)$, and $R_7(G)$. For simplicity, we assume $|R_i| = n; i = 1, ..., 7$, which means each tag in twig consists of n nodes in XML tree. By the linear programming, the size bound of the results of twig query X is n^5 and the size bound of the results of query Q is $n^{\frac{7}{2}}$.

Worst-case optimal algorithm for multi-model join: To achieve the optimality, we expand attributes by satisfying common values and relations of expanding attributes from all databases at the same time. We consider P-C relations of XML twig as a relational table for size bound, but we do not physically transform them into relational tables. We guarantee all the intermediate results are the solution for the subset attributes that have already been expanded. Algorithm 1 illustrates the worst-case optimal algorithm called *XJoin* for multi-model join query.

Example 3.4. Consider the the example in Figure 3. Given relational tables $R_1(A, B, C, D)$, $R_2(E, F, G, H)$ and twig $X(A, B, C, D, E, D, F, G, H)$, the goal is to find the result of query $Q(A, B, C, D, E, D, F, G, H)$ for joining R_1, R_2 and X. Query Q_1 is only for relational tables while Q_2 is only for XML tree. By the Lemma 3.1, the size bound of the results of Q, Q_1 and Q_2 are n^2, n^2 and n^5 respectively. The baseline algorithm combined the results of query Q_1 and Q_2, which may produce n^5 intermediate records in the worst-case situation. *XJoin*, which consider relational tables and XML trees as a whole, generates no more than n^2 intermediate records in any stage of the algorithm. The last graph in Figure 3 on synthetic data shows that *XJoin* outperforms the baseline algorithm in terms of running time and intermediate result size.

Lemma 3.5. *The intermediate result size in any stage in Algorithm 1 matches the wost-case size bound computed by linear program in Equation 1.*

4 CONTRIBUTION

The expected contributions are as follows: (1) We propose to study a research problem to find the worst-case optimal algorithm for relational and XML data join in a multi-model database; (2) We propose a linear program solution to compute the worst-case size bound for joining relational and XML data; (3) We develop a worst-case optimal join algorithm called *XJoin* to match the size bound. *XJoin* significantly outperforms the baseline join algorithm in terms of running time and intermediate result size.

In the on-going work, we will improve the worst-case algorithm by filtering infeasible intermediate results and partially validating the twig structure during the joining.

ALGORITHM 1: Optimal Join algorithm: XJoin

Input: priority of attributes expansion P_A, XML twigs S_x, relational tables S_r
Output: Join results R
$S \leftarrow S_r \cup transform(S_x)$; // continuous P-C relations from XML
$R \leftarrow \varnothing$; $A \leftarrow \varnothing$; // attributes set A
foreach $p \in P_A$; // Optimality guarantees by Lemma 3.5
do
  Get expanding result E from common value of p in S
  Filter E by satisfying relation between p and A in S
  Expend R by E
  $A \leftarrow A \cup o$; // Expend attributes
end
Filter R by validating structure of S_x
return R

REFERENCES

[1] Shurug Al-Khalifa, HV Jagadish, Nick Koudas, Jignesh M Patel, Divesh Srivastava, and Yuqing Wu. 2002. Structural joins: A primitive for efficient XML query pattern matching. In *Data Engineering, 2002. Proceedings. 18th International Conference on*. IEEE, 141–152.

[2] Albert Atserias, Martin Grohe, and Dániel Marx. 2008. Size bounds and query plans for relational joins. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*. IEEE, 739–748.

[3] Jiaheng Lu. 2017. Towards Benchmarking Multi-Model Databases.. In *CIDR*.

[4] Jiaheng Lu and Irena Holubová. 2017. Multi-model Data Management: What's New and What's Next?. In *Proceedings of the 20th International Conference on Extending Database Technology, EDBT 2017, Venice, Italy, March 21-24, 2017*. 602–605.

[5] Jiaheng Lu, Tok Wang Ling, Chee-Yong Chan, and Ting Chen. 2005. From region encoding to extended dewey: On efficient processing of XML twig pattern matching. In *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 193–204.

[6] Jiaheng Lu, Zhen Hua Liu, Pengfei Xu, and Chao Zhang. 2016. UDBMS: Road to Unification for Multi-model Data Management. *CoRR* abs/1612.08050 (2016).

[7] Hung Q Ngo, Ely Porat, Christopher Ré, and Atri Rudra. 2012. Worst-case optimal join algorithms. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*. ACM, 37–48.

[8] Hung Q Ngo, Christopher Ré, and Atri Rudra. 2014. Skew strikes back: New developments in the theory of join algorithms. *ACM SIGMOD Record* 42, 4 (2014), 5–16.

[9] Todd L Veldhuizen. 2012. Leapfrog triejoin: A simple, worst-case optimal join algorithm. *arXiv preprint arXiv:1210.0481* (2012).